PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04N 5/00, 7/16

A1

(11) International Publication Number: **WO 00/30345**

(43) International Publication Date: 25 May 2000 (25.05.00)

(21) International Application Number: PCT/US99/23346

(22) International Filing Date: 7 October 1999 (07.10.99)

(30) Priority Data:
60/107,949 12 November 1998 (12.11.98) US

(71) Applicant *(for all designated States except US)*: GENERAL INSTRUMENT CORPORATION [US/US]; 101 Tournament Drive, Horsham, PA 19044 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: PETERKA, Petr [US/US]; 5126 Caminito Vista Lujo, San Diego, CA 92130 (US). MEANDZIJA, Branislav, N. [US/US]; 827 Coast Boulevard South, La Jolla, CA 92037 (US).

(74) Agent: LIPSITZ, Barry, R.; Building No. 8, 755 Main Street, Monroe, CT 06468 (US).

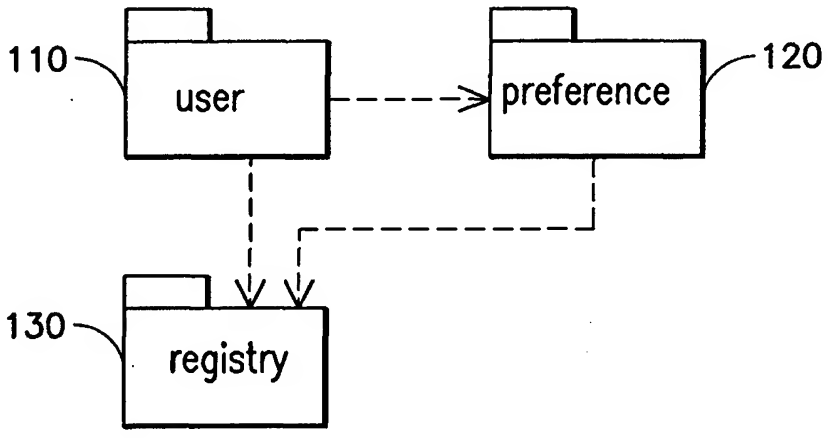
(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: DIGITAL TELEVISION RECEIVER WITH APPLICATION PROGRAMMING INTERFACE FOR USER MANAGEMENT

(57) Abstract

An Application Programming Interface (API) for applications to manage/access user–related information on a Digital Television (DTV) Receiver/Terminal. The API provides a multi–user environment (using user registry (210) and user profile (220) classes), and a registry for preferences (310) which can be associated with an individual user (user–specific) or be common to all users (system–wide). A set of permissions (230) is associated with each user, or at least with a default user, which represents anybody watching the TV. The permission may provide, for example, a parental control/rating ceiling or permission to buy Impulse Pay Per View (IPPV) programs, or engage in e–commerce transactions (e.g., purchase goods or services via the television). The invention also supports a mechanism (220) where some applications can, and some cannot, access the preferences based on a security policy. This means that the user may allow only specified trusted applications to access his/her credit card number but not others, for example, for an e–commerce application.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

DIGITAL TELEVISION RECEIVER WITH APPLICATION PROGRAMMING INTERFACE FOR USER MANAGEMENT

BACKGROUND OF THE INVENTION

This application claims the benefit of U.S. Provisional Application No. 60/107,949, filed November 12, 1998.

The present invention provides an Application Programming Interface (API) for downloadable broadcast applications to manage/access user-related information on a Digital Television (DTV) Receiver/Terminal.

A set-top terminal, also referred to as an Integrated Receiver-Decoder (IRD) or a subscriber terminal, is a device that receives and decodes television signals for presentation by a television. The signals can be delivered over a satellite, through a cable plant, or by means of terrestrial broadcast, for example. Various applications have been proposed, or are currently available, via modern set tops, including video on demand (VOD), audio on demand, pay-per-view, interactive shopping, electronic commerce, electronic program guides, Internet browsers, mail services (e.g., text e-mail, voice mail, audio mail, and/or video mail), telephony services, stock ticker, weather data, travel information, games, gambling, banking, shopping, voting, and others. Applications may also enable Internet connectivity and possibly

Internet-based telephony. The set top functionality is enabled through specialized hardware and software.

The applications may be downloaded by terminals via a network, loaded locally (e.g., via a smart card), or installed at the time of manufacture, for example.

Moreover, with the increasing integration of computer networks such as the Internet, telephony networks, and broadband distribution networks, many opportunities arise for providing new types of applications.

To optimize the user's ability to access these applications, there is a need for user management at the terminals. In particular, it would be desirable to provide a user-management API that can be implemented in software independently of a terminal's hardware and operating system.

The API should allow new users to be recognized, or old users to be deleted, as well as recognizing a current user of the terminal. The API should provide a profile of the each user and maintain a record of permissions that have been granted to the users, e.g., for accessing applications such as pay-per-view, or enforcing rating guidelines for children.

The API should maintain a record of user preferences, such as preferred language and rating ceilings, personal data such as age, address, zip code, account numbers, as well as system-wide preferences, such as favorite channels, language, etc.

The API should also maintain a registry of applications, resources, preferences and users. An application may use/access a resource, which is usually

a device, function or a process on the receiver (e.g. tuner, modem, database, etc.)

The API should be compatible with Java(tm), ActiveX(tm) or an equivalent type of component based object-oriented technology.

The API should be compatible with Digital Audio Visual Council (DAVIC), American Television Standards Committee (ATSC) T3/S17 Digital TV Application Software Environment (DASE), Digital Video Broadcast (DVB)-Multi-Media Home Platform (MHP) and other related environments.

The API should be compatible with any application at a terminal, regardless of how the application was received or installed (e.g., downloaded, resident, from smart card, etc.)

The present invention provides a system having the above and other advantages.

SUMMARY OF THE INVENTION

The present invention provides a software architecture a set-top television terminal. In particular, an Application Programming Interface (API) is provided for applications to manage/access user-related information on the terminal.

A "user" is one who is watching the TV or using its other functions, e.g., E-mail, games, etc.

The API includes a user software package that includes registry, profile, permissions, change cause, change event, and registry event functions. A preferences package includes a registry, names, rating, preferred language, change cause, change event and registry event functions. A registry package includes type, factory, change event, listener, user, preference, resource and application functions.

The present invention provides a multi-user environment (user registry and user profile), and a registry for preferences which can be associated with an individual user (user-specific) or be common to all users (system-wide). A set of permissions is associated with each user, or at least with a default user, which represents anybody watching the TV.

In a particular embodiment, a television set-top terminal includes a computer readable medium having computer program code means, and means for executing the computer program code means to implement an Application Programming Interface (API). The API provides (a) a user registry of a plurality of users of the terminal, (b) a preferences registry of preferences

of the users, and (c) permission(s) for controlling the users' access to at least one application that is provided at the terminal.

The API provides a security policy to allow only specified applications to access the preferences registry. The security policy may be user-controlled. Thus, some applications can, and some cannot, access the preferences based on a security policy. This means that the user may allow only specified trusted applications to access his/her credit card number but not others, for example. These applications may be specified shop-at-home channels or the like.

The API disallows access to user preferences by an application that does not have the required permission(s).

The permission(s) may be associated with each user individually, or may be associated with a default user. A default user might be provided to indicate that the entire family is watching the television together, or to indicate a guest or unknown person.

Additionally, the application generally is responsive to the user preferences. For example, if an application such as an Electronic Program Guide (EPG) accesses a language preference of a particular user, the application can automatically select the language for text when the service information (SI) is multilingual. An application that tunes to channels (audio/video or audio-only) may automatically select the appropriate audio language based on the user's language preference.

As another example, an EPG application may access

favorite channel information of a user to prepare a custom made guide to the programs that are currently playing. Or, an application which enables targeted advertisements may access information regarding a user's location (ZIP code), gender, age, family status, and other personal information, such as pets, hobbies, etc.

Additionally, e-commerce-type applications may need to look at user credit card numbers, permissions to do on-line purchases, and other related preferences.

The permission may also provide, for example, a parental control/rating ceiling or permission to buy Impulse Pay Per View (IPPV) programs, or engage in e-commerce transactions (e.g., purchase goods or services via the television - home shopping).

The preferences include a ratings ceiling preference.

Moreover, the user preferences may include system-wide or user-specific.

The user registry can be used to register a new user of the terminal, identify a current user of the terminal, and remove a former user of the terminal.

For example, when an application needs to switch between users, it may do it by identifying the user using a logon screen, by PIN codes or passwords, or even personalized smart cards, or voice control, for example.

The permission(s) enable the users to access a resource/invoke functions of the terminal. This refers to accessing either some physical resource on the receiver, such as a modem or a tuner, the smart card,

etc., or an application, such as e-mail, web browsing, e-commerce, etc.

The resource may be, for example, a device, function or a process on the receiver, such as a tuner, modem, database, plug-in module, cable, software module, network interface card, persistent storage, TV screen space, memory, CPU, conditional access (CA) module, and so forth.

The API also is adapted to define one or more new types of user preferences, and add the new types of user preferences to the preferences registry. The new user preferences can be defined by a base interface, Preference, which is an object that can be added or removed from the PreferenceRegistry. This allows one to define new types of user preferences, e.g., by extending it the same way as the LanguagePreference extends the base Preference.

For example, the preferences may include specific types of programs (sports, drama, humor, action, etc.).

The API also is adapted to associate the new types of user preferences with the users.

The API may be independent of an operating system and hardware of the terminal.

A corresponding method is also presented.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 shows package relationships and dependencies in accordance with the present invention.

FIG. 2 illustrates a user class/interface diagram in accordance with the present invention.

FIG. 3 illustrates a preferences class/interface diagram in accordance with the present invention.

FIG. 4 illustrates a registry class/interface diagram in accordance with the present invention.

DETAILED DESCRIPTION OF THE INVENTION

1. INTRODUCTION

The present invention provides an Application Programming Interface (API) for downloadable broadcast applications to manage/access user-related information on a Digital Television (DTV) Receiver.

The invention comprises the following two API components: 1) User API; and 2) User Preferences API.

They can be used together where each user has a set of user preferences associated with it, or separately where there is a system-level set of preferences without a user association.

A Registry package is also described herein, which provides a generic mechanism for registry-type constructs.

2. MODEL DESCRIPTION

FIG. 1 illustrates an API package in accordance with the present invention, namely a User package 110, a Preferences package 120, and a Registry package 130. These packages are described in the following sections, in connection with FIGs 2-4.

Note that portions of the disclosure were generated automatically from Rational Rose(tm) CASE tool, developed by Rational Software Corporation, USA. The figures use the Rational Rose (tm) depiction of the Unified Modeling Language (UML), which is a language for specifying, constructing, visualizing, and documenting the artifacts of a software-intensive system. A class diagram represents the static

structure of a system, and shows a pattern of behaviors that the system exhibits. This is accomplished by showing the existence of classes and their relationships. Each class is represented by a box with three sections. The top section lists the class name. The middle section denotes a list of attributes, and the bottom section denotes a list of operations.

A solid or dashed line between classes denotes an association or dependency. A white diamond tip denotes aggregation by reference, while a black diamond tip denotes aggregation by value. A triangular arrowhead denotes a restricted navigation, e.g., inheritance of operation but not of structure.

Moreover, interfaces and classes begin with an uppercase letter, while methods begin with a lowercase letter. A class is a template that defines a data structure, method and function calls for an object. An interface defines a set of methods/function calls that can be manipulated by a class. The class provides the code for implementing an interface.

2.1 User Package

FIG. 2 illustrates a user class/interface diagram in accordance with the present invention. A main UserRegistry interface 210 is the access point for getting information about any user. One or more users can be associated with each terminal (e.g., DTV receiver, set-top box, IRD, TV-enabled PC, etc.) Existing users can be retrieved, new ones can be created, and the current active user can be set via the UserRegistry object 210. The user is represented by the

UserProfile object 220, which holds the user's name and the user's preferences. The preferences may be that the user prefers to view/use certain types of television programs (e.g., favorite channels) and/or applications, preferred audio language, personal data, such age, location (address/zip code), credit card numbers, etc. The user can be authenticated by invoking the implementation-neutral method authenticate(), which may invoke the particular mechanism of authenticating a user, such as Personal Identification Number (PIN) codes, passwords, etc. Users are associated with permissions which are internally used to enforce a security policy. Therefore, new permissions can be granted to each user. The permissions are implementation-specific since they are internal to the terminal.

2.2 Preferences Package

FIG. 3 illustrates a preferences class/interface diagram in accordance with the present invention.

The preference package 120 serves two primary purposes: to hold system-wide preferences and to hold user-specific preferences.

A system-wide preference is anything that applies to all users. For example, English may be a system wide preferred language which applies to all user unless they override it with their own user-specific preference (e.g., Spanish).

When the preference is used in the system-wide sense, the access to it is provided via the PreferenceRegistry 310 obtained from the

RegistryFactory 430 (FIG. 4). The user-specific preferences can be obtained directly from the user (UserProfile 220).

Any new preferences or settings must derive from the abstract Preference interface 330, which provides the listener mechanism and access to the unique preference name. The derived class/interface can provide any methods, behavior and data structures needed to describe the specific preference or settings object.

Each application can register as a listener (using the addPropertyChangeListener() method) to a specific preference and be notified via the propertyChange() method it must implement.

Properties are uniquely identified by their names. If an application tries to store a Property with a duplicate name, the PreferenceAlreadyExists Exception 390 will be thrown. If the application does not have enough privileges to perform any of the PreferenceRegistry 310 methods, the AccessDeniedException 385 will be thrown.

PropertyChangeEvent 350 is defined in the standard Java packages (JDK 1.2). Also, the notations "from security" in class 385, and "from beans" refer to classes ("Security" and "Beans", respectively) that are defined in JDK 1.2.

2.3 Registry Package

FIG. 4 illustrates a registry class/interface diagram in accordance with the present invention.

The Registry package 130 provides a basic mechanism to construct a Registry object of any kind. The Registry interface 410 is a base interface which is extended by all specific Registries, such as the UserRegistry 210 or the PreferenceRegistry 310.

A RegistryListener interface 440 and a RegistryChangeEvent interface 250 are associated with this package. The listener interface 440 is used by any object that wants to be notified of any changes in the Registry 410. Changes are considered those that affect the Registry 410 itself (not necessarily the individual elements in the registry), such as adding or removing elements to/from the Registry. The RegistryChangeEvent 250 is an abstract class which is extended by the specific registry events.

Since most of the API is defined in terms of Java Interfaces, the RegistryFactory 430 is a class that hides the actual object construction implementation.

The notation "from util" in the EventObject class 415 refers to a class "Util" that is defined in JDK 1.2.

The notation "from resource" in the ResourceRegistry interface 480 refers to a class "Resource" that is discussed in commonly-assigned PCT Patent Application No. _____, filed October 7, 1999, and entitled "Application Programming Interface (API) For Accessing And Managing Resources In A Digital Television Receiver."

The notation "from application" in the ApplicationRegistry interface 490 refers to a class "Application" that is discussed in commonly-assigned

PCT Patent Application No. _____, filed October 7, 1999, and entitled "Software Application Lifecycle And Management For Broadcast Applications."

3. CLASS AND INTERFACE DESCRIPTION

The following sections provide the details of each class, interface and its methods.

3.1 User Package

This package 110 provides classes and interfaces necessary for user management functions.

3.1.1 UserRegistry

UserRegistry interface 210 provides access to all users defined on the system. It is derived from Registry 130.

Public Operations:

createUser (name : String) : UserProfile

This method will create a new user of the specified name.

Public operations are those methods that may be called and used by other objects since they are visible outside of the object (e.g., class). In contrast, private operations are visible only to the class itself.

getCurrentUser () : UserProfile

This method returns the user who is currently registered as the active user.

getUserNames () : String[]

This method returns a list of all known users in the registry.

setCurrentUser (newUser : String) : UserProfile

This method defines a new current user. The implementation may prompt the user for a password or some other method of authentication.

getUser (name : String) : UserProfile

This method returns a UserProfile object of the specified name.

3.1.2 UserProfile

This interface 220 represents a container of a single user information, such as settings and preferences, billing info, etc. it is derived from UserPermissions 230.

Public Operations:

getName () : String

Returns the name of this user.

getPreferences () : PreferenceRegistry

This method returns this user's PreferenceRegistry. All operations performed on the returned list of preferences will be reflected in this UserProfile.

authenticate () : boolean

This method is called to authenticate a user. It invokes an implementation-specific mechanism for user authentication. It may use a user dialog to ask for a password or PIN, or other authentication mechanisms.

Returns TRUE if authentication succeeded; FALSE otherwise.

grantPermission (newPermission : String) : void

This method is called to add a new permission to the user. See UserPermissions.

3.1.3 UserPermissions

This interface 230 defines a list of user permissions that can be granted to a user.

Public Attributes:

IPPV : String = "IPPV Purchase"

RATING : String = "Rating Override"

3.1.4 UserRegistryEvent 260

Derived from RegistryChangeEvent 250.

Public Operations:

getUserName () : java.lang.String

This method returns the User Name of the User that caused this event.

3.1.5 UserChangeCause

This interface 240 defines possible causes for the UserRegistryEvent 260.

Public Attributes:

USER_ADDED : short = 1

USER_REMOVED : short = 2

NEW_CURRENT_USER : short = 3

3.2 Preferences Package

This package 120 defines a set of interfaces and classes which provide a mechanism to define a set of preferences, either at the system level or at the user level.

3.2.1 PreferenceAlreadyExistsException

This exception 390 signals that a preference of the same name is already present in the registry and cannot be inserted (added) again. It is derived from Exception, which is in the JDK 1.2 core Java packages - java.lang.Exception.

3.2.2 PreferenceRegistry

This interface 310 represents a registry of all settings and preferences that can be shared by multiple applications. It is derived from Registry 130.

Public Operations:

getPreference (preferenceName : String) : org.atsc.preferences.Preference

This method returns the preference of the specified name, or null if the name is unknown to the registry.

addPreference (aPreference : org.atsc.preferences.Preference) : void

This method allows an application to insert a new preference object into the registry. The new preference must be of a unique name.

removePreference (preferenceName : String) : void

This method allows an application to remove a preference object from the registry.

listPreferences () : org.atsc.preferences.Preference[]

This method returns a list of all Preferences currently stored in the registry.

3.2.3 PreferredLanguage

This interface 340 is an example of a language preference interface. An ordered list of ISO 639.2 alpha-3 strings is used. The first language in the list is the most desirable language. It is derived from the Preference interface 330.

Public Operations:

getLanguage () : String[]

This method returns an ordered list (most desirable first) of three letter ISO language codes.

setLanguage (valueList : String[]) : void

This method allows an application to change the language preference. It returns the new list.

3.2.4 RatingPreference

This interface 335 represents a parental rating preference based, e.g., on age. It is derived from the Preference interface 330.

Public Operations:

getRatingCeiling () : String

This method returns the maximum rating value allowed for watching.

setRatingCeiling (aValue : String) : String

This method allows an application to change the rating ceiling. It returns the new rating level.

3.2.5 Preference

This is a top-level interface 330 which is common for all preference/settings subinterfaces. It supports the listener model and provides the preference name.

This interface can be extended to support specific preferences with specific access methods. It is derived from PreferenceNames 320.

Public Operations:

addPropertyChangeListener (aListener : PropertyChangeListener) : void

This method allows applications interested in changes to this preference to register for preference change events.

removePropertyChangeListener (aListener : PropertyChangeListener) : void

This method allows a preference change listener to remove itself from the list of listeners.

getPreferenceName () : String

This method returns a unique preference name.

3.2.6 PreferenceNames

This interface 320 contains a list of predefined preference names.

Public Attributes:

SIMPLE_RATING : String = "Simple Rating"

LANGUAGE : String = "Language"

3.2.7 PreferenceRegistryEvent

This event 360 informs the RegistryListener 440 about changes in the PreferenceRegistry 310. It is derived from RegistryChangeEvent 250.

Public Operations:

getPreference () : org.atsc.preferences.Preference

Returns the preference that has changed in the repository. This change concerns the repository, such

as adding or removing a Preference from the PreferenceRegistry 310, not a change in the value of the Preference.

3.2.8 PreferenceChangeCause

This interface 345 defines reasons for a PreferenceRegistryEvent 360.

Public Attributes:

PREFERENCE_ADDED : short = 1

PREFERENCE_REMOVED : short = 2

3.3 Registry Package

This package 130 provides a set of supporting and utility classes and interfaces used by other packages.

3.3.1 Registry

This interface 410 provides a common root to all specialized registry interfaces, such ApplicationRegistry 490, ResourceRegistry 480, PreferenceRegistry 310, etc. It is provided so that the RegistryFactory 430 can return a base type.

A "base type" is known from the field of object-oriented programming. To illustrate, one can define a class with a set of functions (methods) and internal variables (e.g., a class "Fruit" which represents fruit and its basic characteristics). One can specialize it by defining a new class, "Apple", which inherits everything from the class "Fruit", and adds new functions that are applicable only to Apples but not to Fruit in general. "Fruit" is then referred to as a "base class" or a "base type."

The Registry interface 410 is derived from RegistryType 405.

Public Operations:

getRegistryType () : String

Called to determine the type of registry implemented by the object returned by the RegistryFatory's (430) method getRegistry().

addRegistryListener (listener: RegistryListener) :

Called to register for events generated by the Registry 410.

removeRegistryListener (listener : RegistryListener) :

Called to de-register for events generated by the Registry 410.

3.3.2 RegistryFactory

This class 430 provides a mechanism to create objects that implement specific Registry interfaces, such as the ApplicationRegistry 490. This class is modeled after the Factory Method design pattern, which, as is known from the field of object-oriented programming, is a methodology and structure for solving a problem.

Public Operations:

RegistryFactory () :

Constructor

getRegistry (registryName : String) : org.atsc.registry.Registry

Returns an instance of an object which implements the specified registry interface. Returns null when specified registry does not exist or cannot be created.

The type of the returned object will be one of the derived Registry types, such as the ApplicationRegistry 490.

3.3.3 RegistryType

This interface 405 defines names for different registry types, such as an application registry, etc.

Public Attributes:

APPLICATION_REGISTRY : String = "Application Registry"

RESOURCE_REGISTRY : String = "Resource Registry"

PREFERENCE_REGISTRY : String = "Preference Registry"

USER_REGISTRY : String = "User Registry"

3.3.4 RegistryListener

This interface 440 allows an object to listen to changes made to the Registry 410.

Public Operations:

registryChange () : ApplicationRegistryEvent

This method of all registered ApplicationRegistryListeners is called by the ApplicationRegistry object 490 when an ApplicaionRegistryEvent is fired (emitted).

3.3.5 RegistryChangeEvent

This event 250 is a generic registry change event that is extended by all specific registries (such as ApplicationRegistry 490, etc.) to provide specific information about the change. It is derived from EventObject 415.

Public Operations:

getRegistryType () : java.lang.String

Returns the type of a registry that this event is associated with.

getCause () : short

Returns the cause of the RegistryChangeEvent 250. Each derived event will define a set of causes appropriate for the registry it represents.

Accordingly, it can be seen that the present invention provides an API for applications to manage/access user-related information on a Digital Television (DTV) Receiver/Terminal. The API provides a multi-user environment, and a registry for preferences which can be associated with an individual user or be common to all users. A set of permissions is associated with each user, or at least with a default user. The permission may provide, for example, a parental control/rating ceiling or permission to buy Impulse Pay Per View (IPPV) programs, or engage in e-commerce transactions (e.g., purchase goods or services via the television). The invention also supports a mechanism where some applications can, and some cannot, access the preferences based on a security policy. This means that the user may allow only specified trusted applications to access his/her credit card number but not others, for example, for an e-commerce application.

Although the invention has been described in connection with various specific embodiments, those skilled in the art will appreciate that numerous

adaptations and modifications may be made thereto without departing from the spirit and scope of the invention as set forth in the claims.

For example, while various syntax elements have been discussed herein, note that they are examples only, and any syntax may be used.

Moreover, the invention is suitable for use with virtually any type of network, including cable or satellite television broadband communication networks, local area networks (LANs), metropolitan area networks (MANs), wide area networks (WANs), internets, intranets, and the Internet, or combinations thereof.

Additionally, known computer hardware, firmware and/or software techniques may be used to implement the invention.

What is claimed is:

1. A television set-top terminal, comprising:

a computer readable medium having computer program code means; and

means for executing said computer program code means to implement an Application Programming Interface (API) for accessing user-related information at the terminal, wherein:

the API provides: (a) a user registry of a plurality of users of the terminal, (b) a preferences registry of preferences of the users, and (c) permission(s) for controlling the users' access to at least one application that is provided at the terminal.

2. The terminal of claim 1, wherein:

the API provides a security policy to allow only specified applications to access the preferences registry.

3. The terminal of claim 2, wherein:

the security policy is user-controlled.

4. The terminal of claim 1, wherein:

the permission(s) are associated with each user.

5. The terminal of claim 1, wherein:

the permission(s) are associated with a default user.

6. The terminal of claim 1, wherein:

the at least one application is responsive to the user preferences.

7. The terminal of claim 1, wherein:
the permission(s) control the users' access to a plurality of applications that are provided at the terminal.

8. The terminal of claim 1, wherein:
the preferences include a language preference.

9. The terminal of claim 1, wherein:
the preferences include a ratings ceiling preference.

10. The terminal of claim 1, wherein:
said user preferences include system-wide preferences.

11. The terminal of claim 1, wherein:
said user preferences include user-specific preferences.

12. The terminal of claim 1, wherein:
the user registry registers a new user of the terminal.

13. The terminal of claim 1, wherein:
the user registry identifies a current user of the terminal.

14. The terminal of claim 1, wherein:

the user registry removes a former user of the terminal.

15. The terminal of claim 1, wherein:

the permission(s) enable the users to access a resource of the terminal.

16. The terminal of claim 1, wherein:

the API is adapted to define at least one new type of user preference, and add the new type of user preference to the preferences registry.

17. The terminal of claim 16, wherein:

the API is adapted to associate the new type of user preference with the users.

18. The terminal of claim 1, wherein:

the API disallows access to user preferences by an application that does not have the required permission(s).

19. A method for implementing a software architecture for a television set-top terminal, comprising the steps of:

providing a computer readable medium having computer program code means; and

executing said computer program code means to implement an Application Programming Interface (API) to provide:

(a) a user registry of a plurality of users of the

terminal, (b) a preferences registry of preferences of the users, and (c) permission(s) for controlling the users' access to at least one application that is provided at the terminal.

20. The method of claim 19, wherein:

the API provides a security policy to allow only specified applications to access the preferences registry.

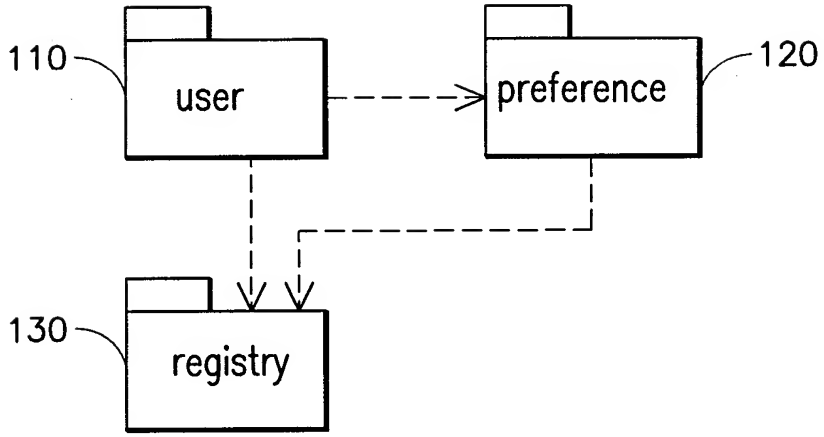


FIG. 1

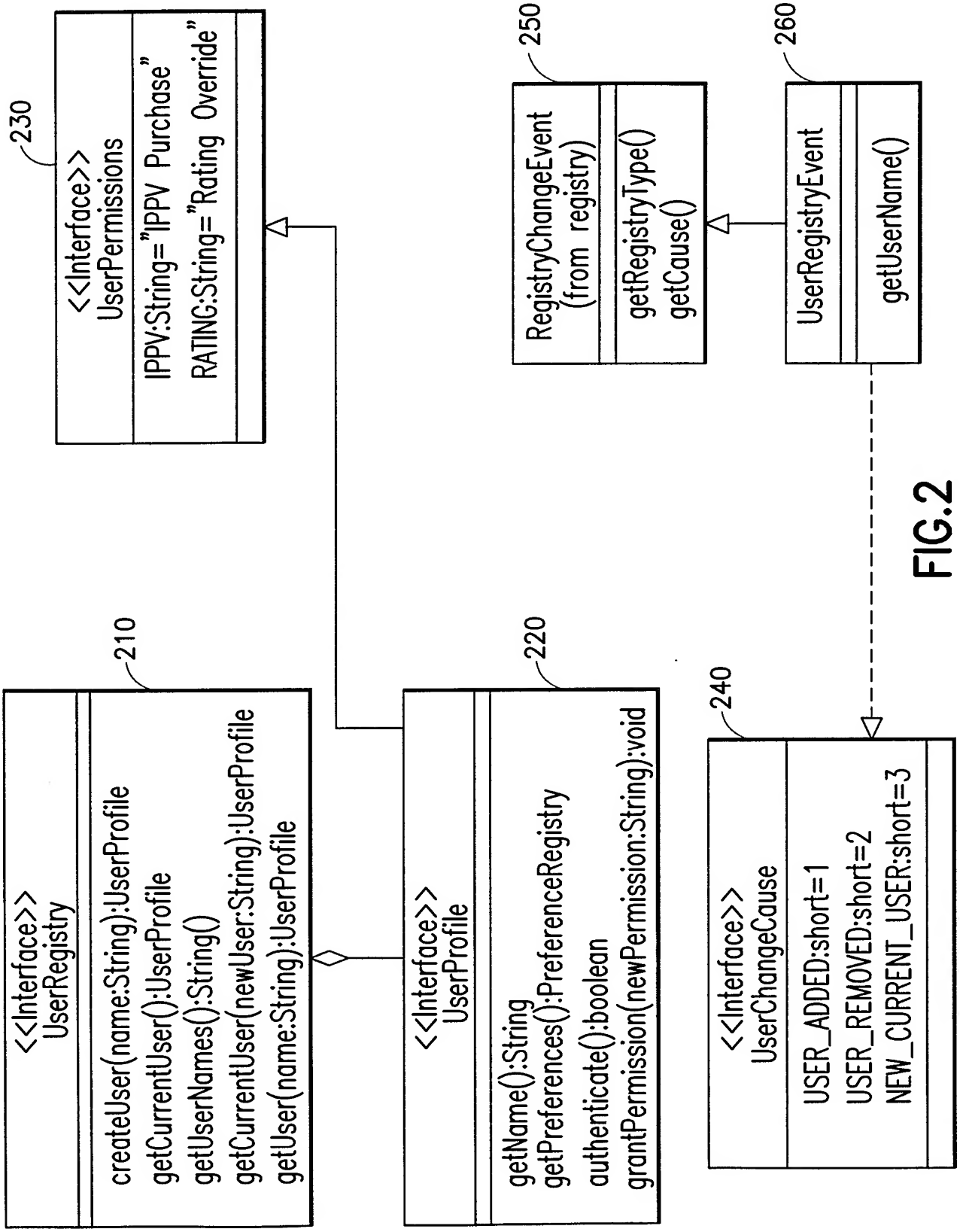

230
<<Interface>>
UserPermissions
IPPV:String="IPPV Purchase"
RATING:String="Rating Override"
250
RegistryChangeEvent
(from registry)
getRegistryType()
getCause()
260
UserRegistryEvent
getUserName()
210
<<Interface>>
UserRegistry
createUser(name:String):UserProfile
getCurrentUser():UserProfile
getUserNames():String()
getCurrentUser(newUser:String):UserProfile
getUser(name:String):UserProfile
220
<<Interface>>
UserProfile
getName():String
getPreferences():PreferenceRegistry
authenticate():boolean
grantPermission(newPermission:String):void
240
<<Interface>>
UserChangeCause
USER_ADDED:short=1
USER_REMOVED:short=2
NEW_CURRENT_USER:short=3

FIG.2

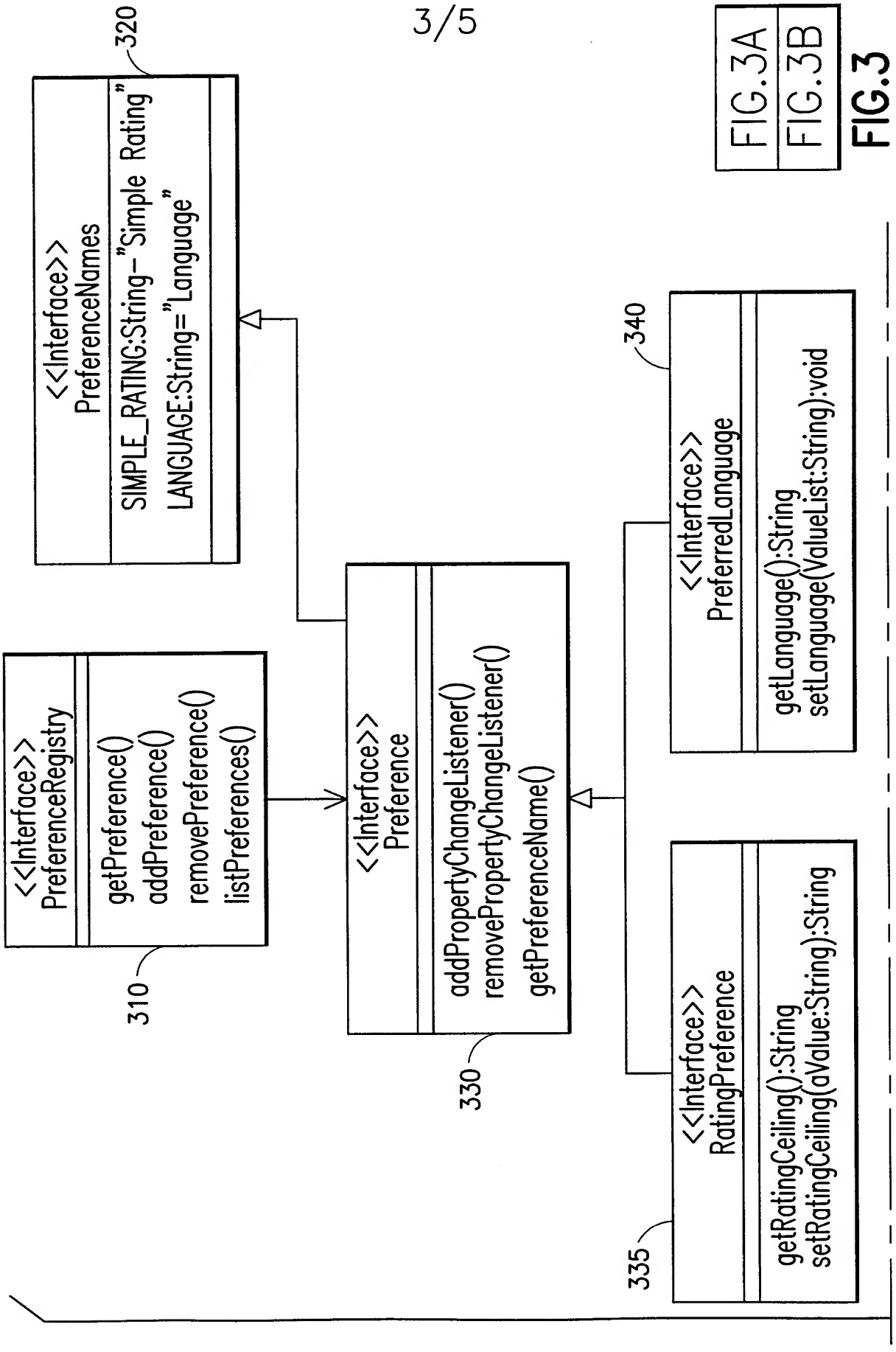


FIG.3A

FIG.3B

FIG.3

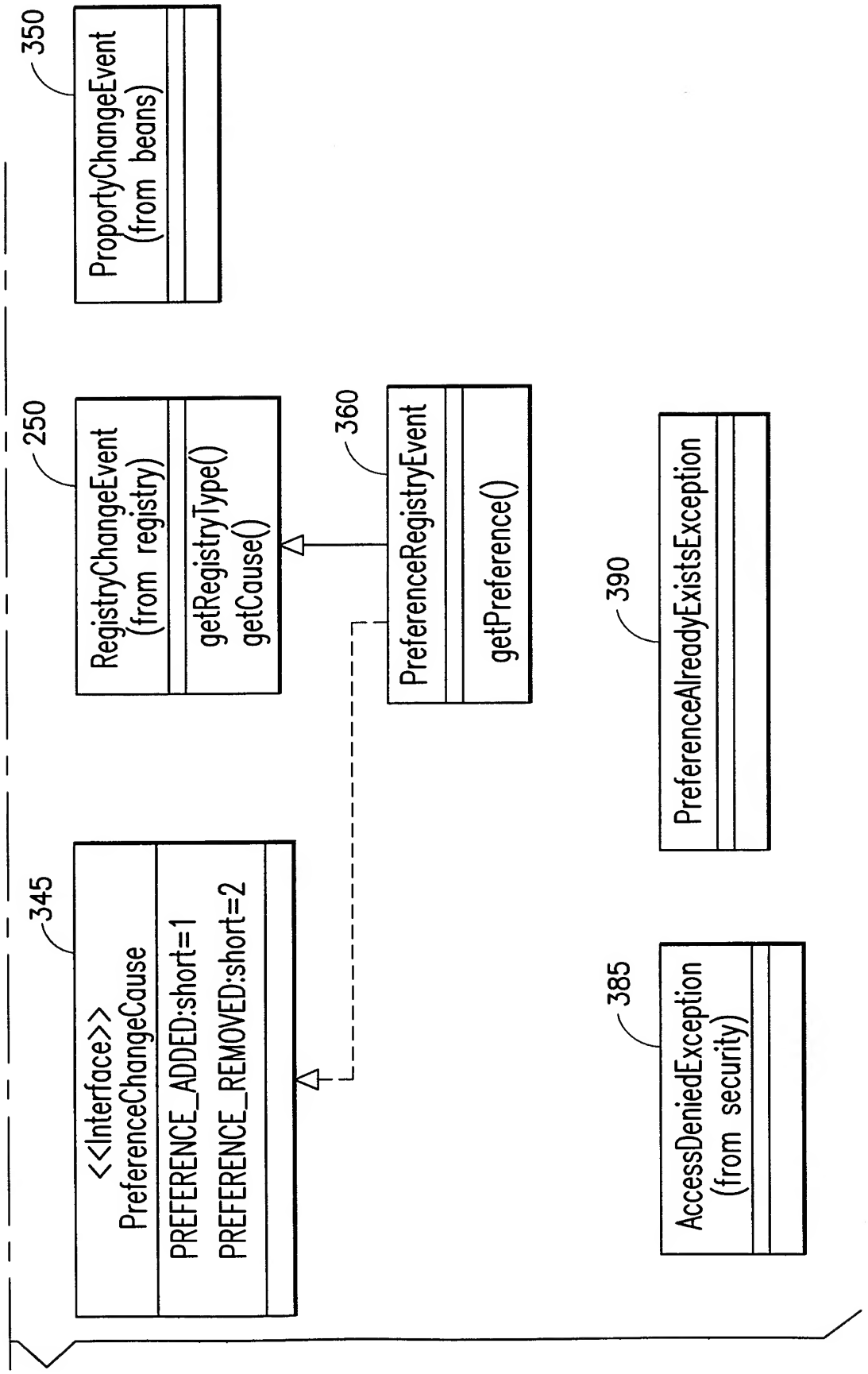

350
ProportyChangeEvent
(from beans)
250
RegistryChangeEvent
(from registry)
getRegistryType()
getCause()
360
PreferenceRegistryEvent
getPreference()
390
PreferenceAlreadyExistsException
345
<<Interface>>
PreferenceChangeCause
PREFERENCE_ADDED:short=1
PREFERENCE_REMOVED:short=2
385
AccessDeniedException
(from security)

FIG.3B

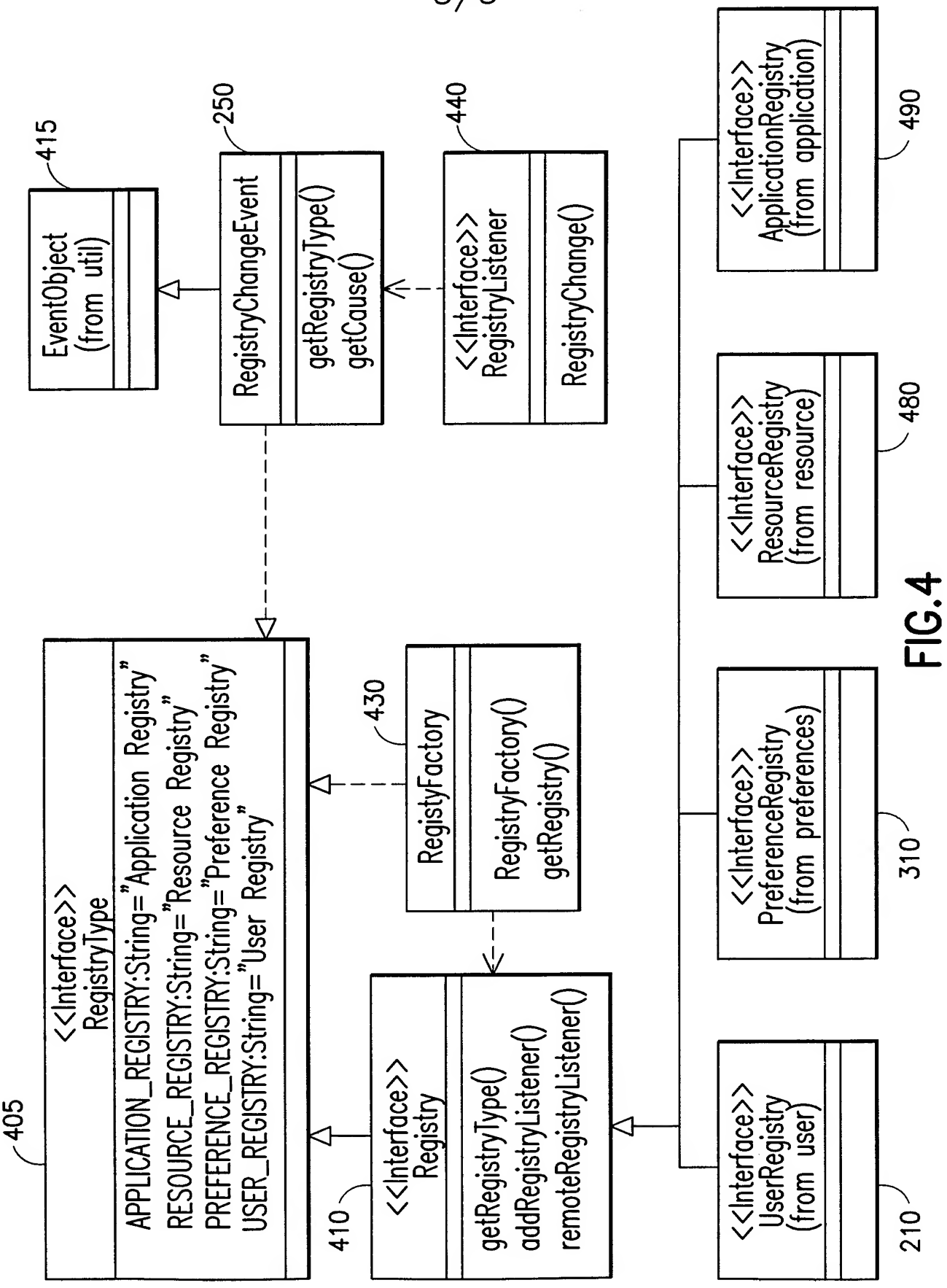

415
EventObject
(from util)
250
RegistryChangeEvent
getRegistryType()
getCause()
440
<<Interface>>
RegistryListener
RegistryChange()
<<Interface>>
ApplicationRegistry
(from application)
490
<<Interface>>
ResourceRegistry
(from resource)
480
405
<<Interface>>
RegistryType
APPLICATION_REGISTRY:String="Application Registry"
RESOURCE_REGISTRY:String="Resource Registry"
PREFERENCE_REGISTRY:String="Preference Registry"
USER_REGISTRY:String="User Registry"
430
RegistryFactory
RegistryFactory()
getRegistry()
410
<<Interface>>
Registry
getRegistryType()
addRegistryListener()
removeRegistryListener()
<<Interface>>
PreferenceRegistry
(from preferences)
310
<<Interface>>
UserRegistry
(from user)
210

FIG.4

INTERNATIONAL SEARCH REPORT

Inter onal Application No
PCT/US 99/23346

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04N5/00 H04N7/16

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
P,X	EP 0 944 257 A (CANAL PLUS SA) 22 September 1999 (1999-09-22) paragraphs '0065!-'0087!; figures 5-7	1,19
X	EP 0 854 645 A (TEXAS INSTRUMENTS INC) 22 July 1998 (1998-07-22) abstract column 9, line 12 - line 36 column 10, line 35 -column 12, line 7 claim 8	1-20
A	US 5 758 257 A (HERZ FREDERICK ET AL) 26 May 1998 (1998-05-26) column 45, line 56 -column 46, line 18; figure 9 column 49, line 7 - line 30	1-20

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
20 January 2000	27/01/2000

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Giannotti, P

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

Inter ...nal Application No
PCT/US 99/23346

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0944257 A	22-09-1999	AU 2636099 A	20-09-1999
		WO 9945709 A	10-09-1999
EP 0854645 A	22-07-1998	JP 10207914 A	07-08-1998
		SG 67469 A	21-09-1999
US 5758257 A	26-05-1998	AU 703247 B	25-03-1999
		AU 4410396 A	19-06-1996
		CA 2207868 A	06-06-1996
		EP 0796538 A	24-09-1997
		WO 9617467 A	06-06-1996
		US 5734720 A	31-03-1998
		US 5754938 A	19-05-1998
		US 5754939 A	19-05-1998
		US 5835087 A	10-11-1998